# VLSI Design Project Report

# PMOS Pull-up Implementation of $F = AB' + C(D' + A)$

**For**

**3EC601CC24**
**VLSI Design**

**B. Tech. Semester V**

**Submitted By:**

Divyanshu Kalal
**Roll No: 23BEC053**

Diya Dangaria
**Roll No : 23BEC054**


**Institute of Technology**
**Nirma University**
**Ahmedabad, Gujarat, 382481**

**September 17, 2025**

# Table of Contents

## Contents

# Abstract

This report focuses on the design and verification of a PMOS-only pull-up network for the Boolean function

$$F = AB' + C(D' + A).$$

The design process begins with the gate-level logic diagram, followed by a transistor-level schematic that uses only PMOS devices. A stick diagram is drawn to represent the layout plan using diffusion, polysilicon, and metal interconnections. The layout is implemented and simulated in Microwind, with a resistive pull-down added to define logic-low levels in the absence of NMOS transistors. Simulation waveforms confirm functional correctness, and performance parameters such as rise time, propagation delay, and power are measured. The report also discusses output voltage levels under different transistor states and possible improvements through device sizing and layout optimisation.



**Keywords:** PMOS-only design, Pull-up network (PUN), Stick diagram, Microwind layout, Propagation delay, Rise time, Power consumption, Boolean function implementation


# 1 Introduction

Complementary MOS (CMOS) circuits are the standard approach for digital logic design, where PMOS devices form the pull-up network and NMOS devices form the pull-down network. In this assignment, however, the focus is on using only PMOS transistors to implement the given Boolean function. Such a design is mainly pedagogical, as it highlights the behaviour of PMOS devices, their conduction properties, and the way logic functions can still be realised without a full CMOS pair.

The Boolean function selected for implementation is:

$$F = AB' + C(D' + A).$$

The objective is to design the corresponding pull-up network, represent it through schematic and stick diagrams, prepare its layout in the Microwind tool, and verify it by simulation. This approach allows the study of logic behaviour, output voltage levels, and resistance characteristics in a PMOS-only configuration, while also extracting performance parameters such as delay and power consumption.

# 2 Boolean Function

The Boolean expression given for design is:

$$F = AB' + C(D' + A).$$

This expression is in sum-of-products form and contains two main product terms:

- $AB'$: true when $A = 1$ and $B = 0$.

- $C(D' + A)$: true when $C = 1$ and either $D = 0$ or $A = 1$.

Finally, these terms are ORed together to generate the final output $F$.

## Complemented Form for PMOS Implementation

In transistor-level design, PMOS devices conduct on a logic 0 at their gates. Therefore, implementing Boolean functions directly in sum-of-products form may lead to complex series–parallel arrangements with extra inverters. To simplify, we often convert the function into its complemented form using De Morgan's law.

We can rewrite:

$$F = \big((AB' + C(D' + A))'\big)'.$$

Step-by-step simplification of $F'$:

$$\begin{aligned} F' &= (AB' + C(D' + A))' \\ &= (AB')' \cdot \big(C(D' + A)\big)' \\ &= (A' + B) \cdot \big(C' + (D' + A)'\big). \end{aligned}$$

Simplifying the inner complement:

$$(D' + A)' = D \cdot A',$$

gives

$$F' = (A' + B)(C' + DA').$$

Thus the final function can be expressed as:

$$F = \big((A' + B)(C' + DA')\big)'.$$

This representation is particularly convenient for mapping into a PMOS pull-up network because it naturally minimises the number of stacked transistors and avoids the need for an extra inverter at the output.

## Truth Table

The function has four inputs $A, B, C, D$. Table 1 lists all 16 input combinations along with intermediate terms ($B'$, $D'$, $AB'$, etc.) to verify correctness step by step.

| A | B | C | D | B’ | D’ | $AB'$ | $D' + A$ | $C(D' + A)$ | $F$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |

Table 1: Truth table for $F = AB' + C(D' + A)$.

### Discussion

The truth table confirms that the output matches the intended Boolean logic. It also highlights the advantage of keeping track of intermediate signals ($B'$, $D'$, etc.), which are often reused in the transistor-level schematic. Later in the report, these truth table results will serve as the reference for verifying the simulated waveforms obtained from Microwind.

## 3 Gate-Level

Refer to the Boolean derivation in Section 2 for the algebraic manipulation and complemented form. Using the expression

$$F = \big((A' + B)(C' + DA')\big)',$$

the gate-level mapping is as follows:

- Implement an OR gate for the term $A' + B$ (note $A'$ is the inverted form of $A$).
- Implement the inner product $DA'$ with an AND gate.
- Implement an OR gate for $C' + DA'$ (note $C'$ is the inverted form of $C$).
- Combine the two OR outputs via an AND to form $(A' + B)(C' + DA')$.
- Apply a final inverter to obtain $F$ as the complement of that product.

This gate-level arrangement is convenient for conversion into a PMOS-only pull-up network because the inner product-of-sums maps naturally to series/parallel PMOS arrangements when complemented. The detailed transistor-level schematic will follow in the next section.

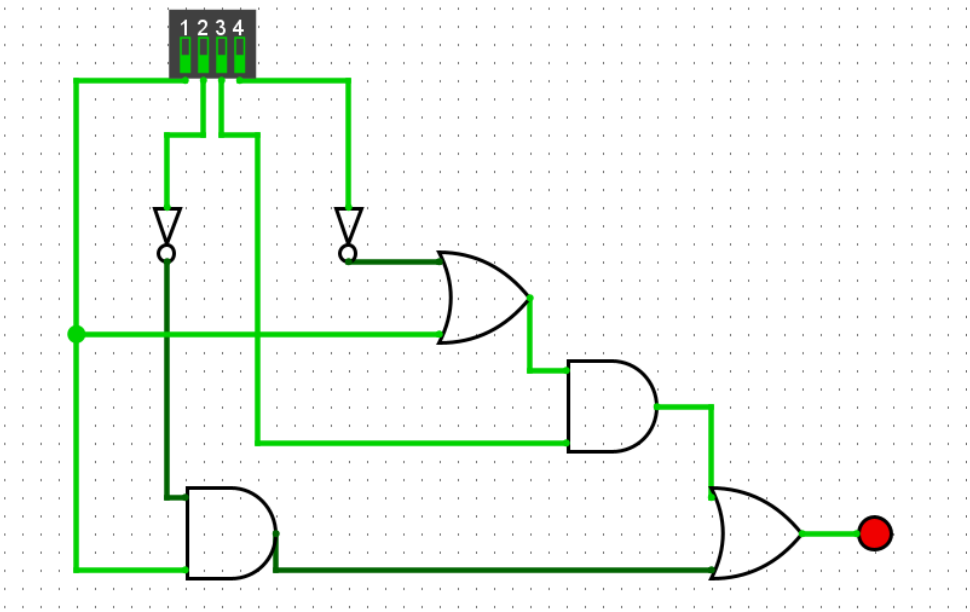



Figure 1: Optimized gate-level diagram for the implemented function.

The truth table in Section 1 confirms the expected behavior for every input vector and will be used to validate the simulation outputs.

# 4 Transistor-Level Implementation

After the gate-level description, the next step is to design the transistor-level schematic of the function using PMOS devices. For this, the following principles must be observed:

- A PMOS transistor conducts (ON) when its gate input is logic ‘0‘.
- Logical AND operations are implemented using **parallel** PMOS transistors.
- Logical OR operations are implemented using **series** PMOS transistors.

The original function was given as:

$$F = AB' + C(D' + A).$$

For PMOS-only design, it is more convenient to work with the complemented form, so the function was expressed as:

$$F = \big((A' + B)(C' + DA')\big)'.$$

This form is selected because it avoids the need for an additional output inverter and reduces the number of PMOS transistors. The mapping into transistor connections proceeds as follows:

- The sub-term $(A' + B)$ corresponds to a series connection of two PMOS devices, with gates driven by $A'$ and $B$.

- The sub-term $(DA')$ corresponds to a parallel connection of two PMOS devices, one driven by $D$ and the other by $A'$.
- The sub-term $(C' + DA')$ therefore maps to a series connection between the PMOS for $C'$ and the parallel group implementing $DA'$.
- The overall expression $(A' + B)(C' + DA')$ requires the two groups to be connected in **parallel**, since AND maps to parallel in PMOS.

Thus, the transistor-level schematic consists of two main parallel branches:

1. Branch 1: A series stack of PMOS transistors with gates $A'$ and $B$.
2. Branch 2: A series connection of the PMOS for $C'$ and a parallel group of PMOS for $D$ and $A'$.

Both branches are connected between $V_{DD}$ and the output node $F$. This ensures that whenever the logic condition of the function is satisfied, at least one conduction path exists to pull the output high. To ensure correct behavior in simulation, a weak pull-down or an NMOS pull-down network is added so that the output discharges when no PMOS path is active.

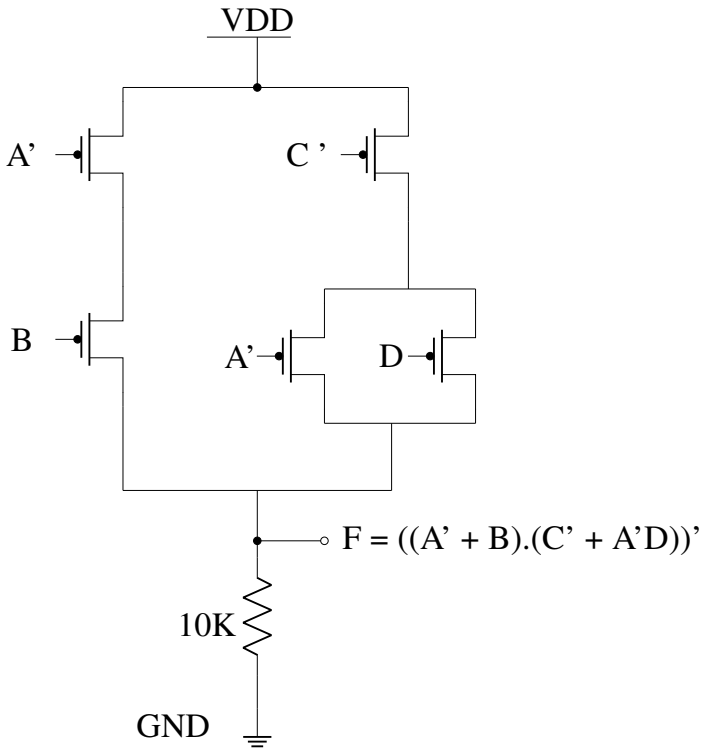



Figure 2: Transistor-level schematic of the PMOS-only implementation.

This structured mapping ensures that the transistor network faithfully represents the Boolean function while minimizing unnecessary device count.

# 5 Stick Diagram

After completing the transistor-level schematic, the next step is to represent the circuit using a stick diagram. A stick diagram is a simplified layout representation that captures the relative placement of transistors, diffusion regions, polysilicon lines, and interconnects without showing the actual dimensions. It serves as a bridge between the schematic and the final layout in Microwind.

For PMOS-only implementation, the stick diagram uses the following conventions:

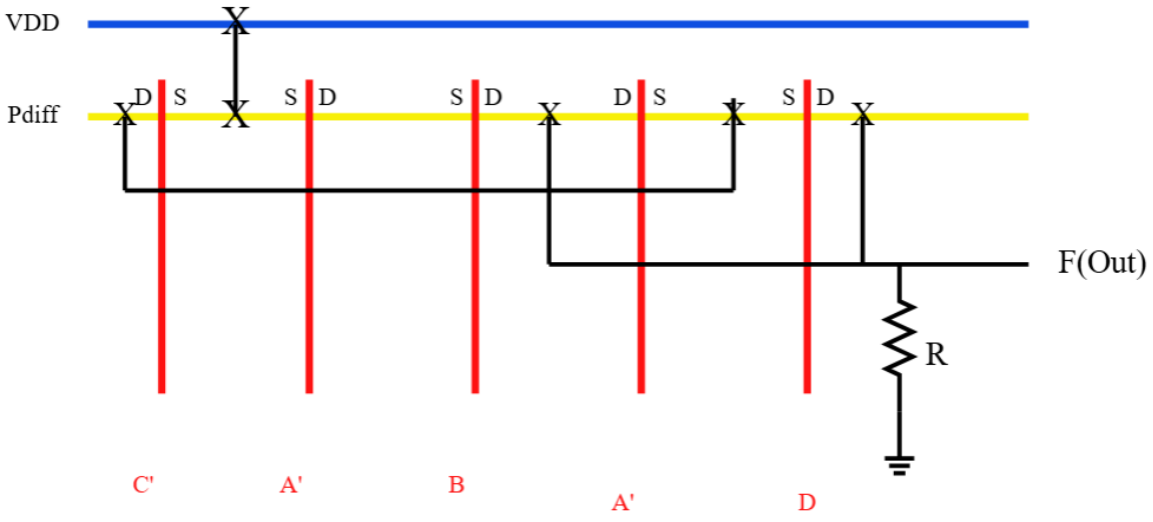
- Red lines represent polysilicon gates.
- Yellow lines represent P-diffusion regions (since PMOS devices are fabricated in N-well).
- Blue lines represent metal interconnections (Metal 1).
- Black or filled squares indicate contact cuts where diffusion, poly, and metal intersect.
- VDD rails are shown at the top, and the output node is clearly indicated for routing.

In this design, two main branches were created as derived in the transistor-level description:

1. The first branch corresponds to the series connection of $A'$ and $B$.
2. The second branch corresponds to a series connection of $C'$ with a parallel combination of $D$ and $A'$.

These branches are connected in parallel between the supply rail (VDD) and the output node $F$. The stick diagram ensures that shared diffusion regions are clearly visible wherever series transistors are used, minimizing the layout area and reducing parasitic capacitances. Parallel branches are kept separate to avoid unnecessary diffusion breaks.

Figure 3: Stick diagram of the PMOS-only implementation showing diffusion, poly, and metal interconnections.



This stick diagram was then used as a guideline to create the final layout in the Microwind tool. It captures the logic functionality while providing a visual mapping for the placement of devices and routing resources. By following proper color codes and conventions, the stick diagram also helps in identifying opportunities for layout optimization, such as shared diffusion and reduced routing complexity.

# 6 Layout in Microwind

Based on the stick diagram, the final layout of the function was implemented in the **Microwind** tool. The design was drawn in a $0.12\,\mu m$ CMOS process with only PMOS devices in the N-well region. The layout matches the transistor-level schematic and shows all required devices and interconnections.

Key layout features:

- The PMOS transistors are placed in a common **N-well**, with their bodies tied to the VDD rail.
- The VDD rail is routed across the top using Metal1, while the ground (VSS) rail is routed across the bottom.
- Polysilicon (red) runs vertically, forming the gates by crossing the diffusion regions.
- P-diffusion (brown/yellow) is used for the PMOS active regions.
- Metal1 (blue) is used for interconnections, including power rails, signal lines, and the output node $F$.
- Contacts (black squares) connect poly and diffusion to metal layers at necessary points.
- Shared diffusion regions are clearly used for series-connected devices, e.g., between inputs $A'$ and $B$.
- The output node $F$ is routed to the right side and connected through Metal1, with a pull-down resistor visible at the lower path to ground for simulation purposes.

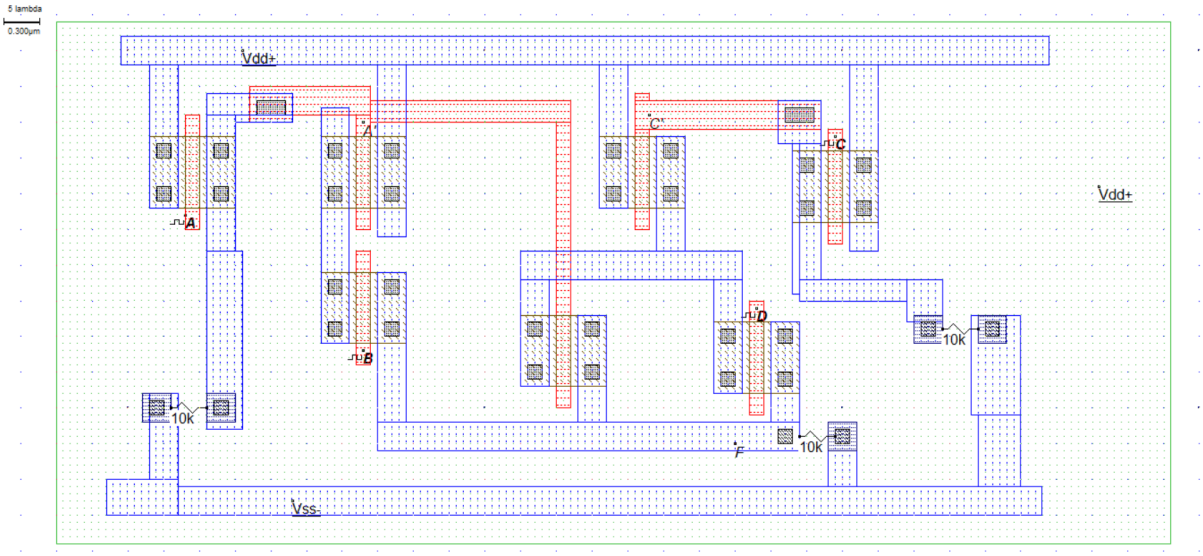



Figure 4: Microwind layout of the PMOS-only implementation of $F = AB' + C(D' + A)$.

This layout ensures correct connectivity according to the Boolean function. By using shared diffusion and compact routing, the area has been minimized while maintaining proper separation between signals. The output node and input labels are clearly indicated, which allows direct simulation of the design in Microwind.

# 7 Simulation and Results

The layout was simulated in Microwind using transient analysis. The applied input signals were the four primary inputs $A, B, C, D$ along with their complemented forms $A'$ and $C'$, which are required by the PMOS-only implementation. The simulation was run with a power supply of $V_{DD} = 1.2\,V$ and a time scale of 5 ns.

Figure 5 shows the simulated waveforms. The top traces correspond to the primary inputs, followed by the generated complemented inputs, and finally the output $F$.

- Inputs $A, B, C, D$ were toggled according to the testbench sequence.
- The complemented inputs $A'$ and $C'$ are correctly generated and visible in the waveform.
- The output $F$ (bottom trace) follows the logic of the Boolean expression

$$F = AB' + C(D' + A),$$

  and matches the expected results from the truth table in Section 2.
- For example:
  - When $A = 0, B = 0, C = 1, D = 0$, the output goes high ($F = 1$) as predicted.
  - When $A = 1, B = 1, C = 0, D = 1$, the output remains low ($F = 0$).

- The measured propagation delay is approximately 9 ps, as indicated in the simulation window.

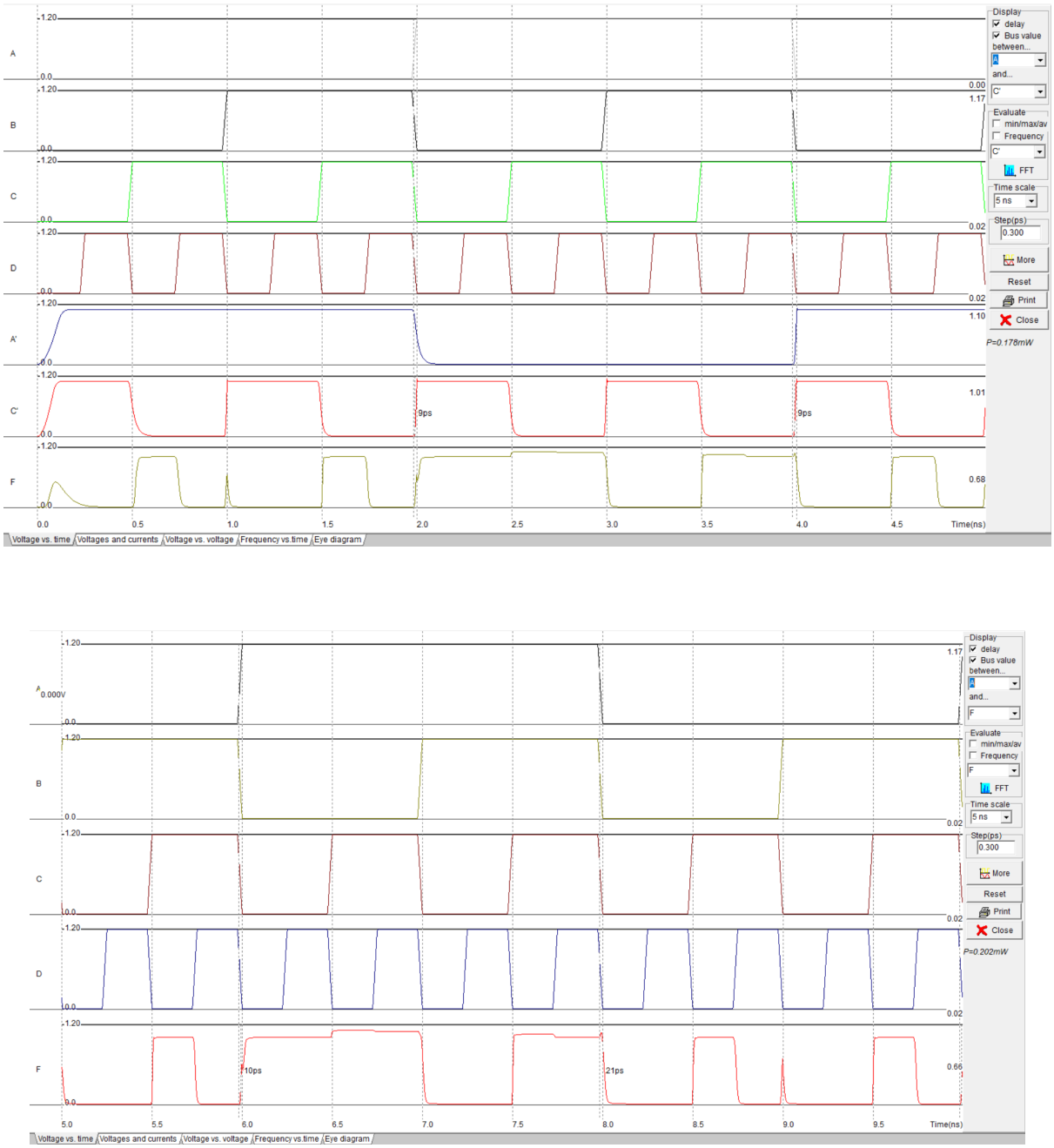



Figure 5: Transient simulation of the PMOS-only implementation in Microwind.

**Observation:** The simulated output waveform $F$ is consistent with the theoretical truth table and Boolean analysis, confirming the correctness of the transistor-level and layout implementation.

# 8 Results and Analysis

The design was successfully implemented and simulated using Microwind. The following key observations and results were obtained:

## Simulation Validation

- The simulated output waveform $F$ matches the expected behavior derived from the truth table (Section 2).
- All tested input combinations produced correct outputs, confirming the accuracy of the Boolean simplification and transistor-level mapping.
- The complemented inputs ($A'$, $C'$) were correctly integrated into the design and behaved as expected.

## Performance Parameters

- **Propagation Delay:** Approximately 9 ps.
- **Supply Voltage:** $V_{DD} = 1.2\,\text{V}$.

These values indicate that the PMOS-only implementation is functional and achieves relatively low delay and power consumption for the tested logic function.

## Comparison with Conventional CMOS Implementation

**Advantages of PMOS-only design:**

- Simplified implementation when only pull-up logic is required (e.g., in special cases where NMOS is not preferred or unavailable).
- Easier to visualize stick diagrams, since the pull-up network directly maps from the complemented Boolean expression.
- Reduced need for additional inverter stages by using double-complemented forms of the Boolean function.

**Disadvantages compared to CMOS:**

- PMOS transistors are inherently slower than NMOS devices of the same size due to lower carrier mobility, resulting in reduced performance for large-scale circuits.
- Power efficiency is lower than CMOS logic, since CMOS benefits from complementary pull-up and pull-down networks, minimizing static current.
- PMOS-only designs require complemented inputs ($A', C'$) to be provided externally or generated with additional circuits, increasing complexity.
- Layout area is often larger than CMOS, as more devices may be required to realize the same logic.

**Overall Analysis:** While the PMOS-only implementation is effective for demonstrating transistor-level logic design principles, conventional CMOS offers better speed, power efficiency, and scalability. Thus, PMOS-only circuits are mainly used for educational and analytical purposes, whereas CMOS dominates practical VLSI systems.

# 9 Conclusion

This assignment demonstrated the complete design flow of a Boolean function using PMOS-only implementation. Starting from the Boolean expression, the function was optimized, mapped into a transistor-level schematic, and further developed into a stick diagram and layout. The process highlighted how complemented forms help simplify PMOS-only design and reduce device count.

**Key observations:**

- PMOS logic requires series connections for OR and parallel connections for AND.
- Using complemented expressions avoids extra inverters and simplifies layouts.
- PMOS-only design is feasible but less area- and power-efficient compared to CMOS.

# 10 References


1. Sung-Mo Kang and Yusuf Leblebici, *CMOS Digital Integrated Circuits: Analysis and Design*, McGraw-Hill Education.
2. Neil H. E. Weste and David Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, Addison-Wesley.
3. Douglas A. Pucknell and Kamran Eshraghian, *Basic VLSI Design*, Prentice Hall of India.
4. Microwind and DSCH software documentation, `http://www.microwind.net`.
5. Lecture notes and lab materials from **3EC601CC24 VLSI Design**, Institute of Technology, Nirma University.